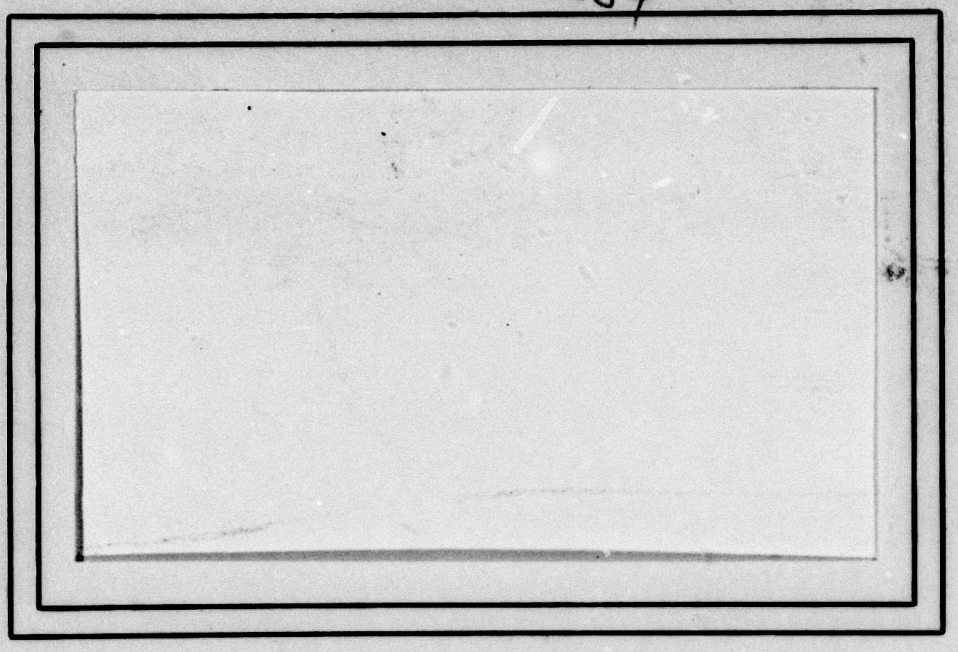


COMPUTER SCIENCE TECHNICAL REPORT SERIES

UNIVERSITY OF MARYLAND
COLLEGE PARK, MARYLAND
20742

(15) DAAG53-76-C-0138,
DARPA Order-3296

(9) Technical rept.,

(12) 26

(14) TR-793
DAAG-53-76C-0138

(11) July 1979

(6) SHAPE SEGMENTATION USING RELAXATION, II.

(10) Wallace S. Rutkowski

Computer Vision Laboratory
Computer Science Center
University of Maryland
College Park, MD 20742

ABSTRACT

The work begun in [1] on the application of relaxation to ambiguous shape segmentation is extended to include curve linking and gap filling. A chain coded input image is broken into segments based on a measure of local curvature. Gap completions linking pairs of segments are then proposed and represented in a graph structure. A second graph, whose nodes consist of paths in the above graph, is constructed, and the nodes of the second graph are probabilistically classified as various object parts. Relaxation is then applied to increase the probability of mutually supporting classifications, and decrease the probability of unsupported decisions. A modified relaxation process using information about the size, spatial position, and orientation of the object parts yielded a high degree of disambiguation.

The support of the Defense Advanced Research Projects Agency and the U.S. Army Night Vision Laboratory under Contract DAAG-53-76C-0138 (DARPA Order 3206) is gratefully acknowledged, as is the help of Kathryn Riley in preparing this paper.

JOB

1. Introduction

There exist many image processing tasks in which the objects of interest in a scene may touch or overlap each other or noise objects. In some instances it is possible to apply simple criteria to correctly segment out the objects of interest in a single pass. In other cases, however, simplistic methods do not work, and some other strategy must be employed. One commonly taken approach is to perform a state space search, in which parts of an object which have been identified serve to guide the further segmentation of the image. This method tends to be both sequential and order dependent. This paper describes a different approach, in which a number of segmentation alternatives are represented initially in a graph structure, and a relaxation process is used to reinforce the correct decisions and prune the poor ones. Specifically, a set of segmentation points is chosen along the object boundary based on a curvature measure [2]. Every piece of boundary curve which lies between two consecutive segmentation points is represented by a node in a graph. A gap filling program [3] then examines the set of curves and adds to the graph both a set of "imaginary" nodes representing gap completions, and a set of arcs linking those nodes representing curve segments which may be consecutive along the boundary of an object part. Our problem then becomes one of tracing a path through the resulting graph which

corresponds to an object [4]. To do this, we trace a multitude of short paths through the graph, and submit the corresponding chain code to a classifier, where it is either rejected or assigned a probability vector, each component of which provides an estimate of the probability that the chain code is a particular part of the object. Those curves which survive classification become nodes in a second level graph structure. Nodes in this second level graph are linked iff their corresponding curve segments are consecutive. Relaxation is then applied to this graph.

2. Construction of the graph structure

This section describes in more detail the manner in which the graph structures mentioned in Section 1 are constructed. It is helpful in describing the process to refer to a simple example (Figure 1). The input to the program consists of a set of closed curves. Each curve is represented as a sequence of (x,y) coordinate pairs. The sequence of coordinates is derived from a four-neighbor chain code. Specifically, if P_i and P_j are two consecutive points in such a sequence, then the vector $P_j - P_i$ is an element of the set {(0,1), (0,-1), (1,0), (-1,0)}. Suppose the input to the program is the single closed curve of Figure 1 (minus the dashed lines). A measure of local curvature is computed at each point of the curve. The measure used was the weighted k-curvature method described in [2]. A subset of the input points is selected as a set of segmentation points. In our case, we performed non-minimum suppression on the curvature values, and then selected as our segmentation points those remaining points whose curvature values were below a preset negative threshold. Thus our segmentation points tend to lie at concave angles. The heavy dots in Figure 1 represent the segmentation points for this example.

Every sequence of consecutive boundary points whose first and last points are segmentation points, and which contains no other segmentation points, is called a segment. A node

is created in our graph structure for each such segment. In Figure 1 we have four segments, labeled A to D.

A linking program then examines this set of segments, and may decide to link various pairs of segments. If the program decides to link segment S_1 to segment S_2, there are two possibilities. If the last point of S_1 is the same as the first point of S_2, then an arc is created in the graph from the node representing S_1 to the node representing S_2. In the example, segments A and B would be linked in this way. If there is a gap between the last point of S_1 and the first point of S_2, then a sequence of points representing a straight line bridging the gap is computed. A new node S_i is created representing this "imaginary" segment, and two arcs are added connecting S_1 to S_i and S_i to S_2. In Figure 1, segment A is linked to segment C via an imaginary segment F.

A graph representing the end result of these operations is shown in Figure 2. Of course, the linking program requires a set of parameters which determine the linking criteria, and the graph which results will depend on these parameters. For example, we could set the parameters so that segments which meet at large concave angles are not linked. In that case the arcs A-B, B-C, C-D, and D-A would not be present. With a different choice of parameters, we could have additional imaginary curves linking A to D, for example, or even A to itself via an imaginary segment in the same position as E.

After the construction of this first level graph, we can view our problem as one of finding a path through this graph which represents the sought for object. This could be done by a typical state space search. This technique, however, tends to be highly sequential and order dependent. We have adopted a different strategy. The object is considered to consist of a number of parts. The program then searches many short paths through the graph, concatenating the curve segments of the nodes along each path and submitting the result to a classifier. (Note that searching a variety of short paths is a local process which could conceivably be carried out by an array of processors working in parallel.) Those paths which survive classification as possible object parts become the nodes in a second level graph. An arc connects a pair of nodes in this graph if their associated curve segments are contiguous. A relaxation process is then run on this graph.

3. Determination of the initial probabilities

Suppose we have a sequence of points $(P_1, P_2, \ldots, P_n)$ representing a curve to be classified. We wish to assign to this curve a probability vector (N,R,T,L) where each component represents the probability that the curve is a nose, right wing, tail, or left wing, respectively. To accomplish this, the program approximates the curve with a two-piece polygon with vertices at P_1, P_k, P_n, where $1 < k < n$ and P_k is chosen to minimize a measure of the error of fit between the polygon and the point sequence. Curves whose polygon approximations fail to meet fixed limits on size, angle, and fit to the original curve are rejected. Any curve not rejected is assigned a probability vector based on a measure of its symmetry (the difference in length between the two sides of the polygon) and the degree of match between polygon and curve. The symmetry is used in the same manner as in [1] to split the probability between three possibilities, nose-tail, left wing, and right wing. The value for nose-tail is then split between the nose and tail possibilities based on the fit error between the curve and its polygonal approximation. In particular, we specify an error limit E_0, at which the probability for the curve being a nose will fall to zero. Then, if the measured error is E, we define:

for $E \geq E_0$, $P(\text{nose}) = 0, P(\text{tail}) = \text{nosetail}$

else $P(\text{tail}) = \frac{E}{E_0} \times \text{nosetail}$

$P(\text{nose}) = \text{nosetail} - P(\text{tail})$

4. The relaxation process

The relaxation process operates on the directed labeled graph described above. During each iteration the probabilities are updated according to the equations below:

Let N = the set of nodes in the graph.

L = the set of labels; in our case,

L = {nose, right wing, tail, left wing}.

A_n = the set of predecessors of node n.

B_n = the set of successors of node n.

$P_m(n,j)$ = the probability of label j on node n at the mth iteration.

Given the $P_m(n,j)$ for all $n \in N$, $j \in L$, we define

$$\max_m(n,j) = \max_{\substack{a\in A \\ b\in B \\ i,k\in L}} P_m(a,i)P_m(n,j)P_m(b,k)\alpha(i,j,k)$$

and then normalize:

$$P_{m+1}(n,j) = \begin{cases} \dfrac{\max_m(n,j)}{\sum_{\ell\in L} \max_m(n,\ell)} & \text{if the denominator is } > 0 \\ 0 & \text{otherwise} \end{cases}$$

A related relaxation scheme, using the product of the probabilities as used here, rather than the more conventional arithmetic average, is discussed in [5].

Intuitively, given a node and a label on that node, we are considering every legal path (of one predecessor and one successor) through that node which uses the label. We compute

a measure of the "goodness" of each such path. We then take the maximum result, since if there is any very good path using this label, we want it to survive with high probability, no matter how many poor paths were investigated.

5. Results

Figure 3 displays pictorially the results of applying this method to the input picture shown in Figure 3a. Remember that in classifying each node a two-piece approximating polygon is constructed. Figures 3b through 3g display the polygons associated with nodes which have survived the indicated number of iterations. After fourteen iterations, the original graph, which consisted of 53 nodes, is pruned to 17 nodes. The pruned graph is illustrated in Figure 4.

The number on each node is its number in the original graph of 53 nodes. The letter is the remaining label assigned to the node. Note that the final graph consists of four disconnected subgraphs. Three of these correspond to the three airplanes, and one is a noise object. The graphs for the second and third planes consist of exactly four nodes each, one for each airplane sub-part. The graph corresponding to the first plane contains two tail nodes. This type of ambiguity can arise because there may be several paths between a given pair of nodes in the first level graph and several of these paths may pass classification as possible features (Figure 5).

The noise object survives because the relaxation process as defined above does not take into account the relative position, size, orientation, etc. of the parts. A triple of labels has compatibility 1 if it is a legal sequence, and 0 otherwise.

We can extend the notion of compatibility, however, to take account of the relationships between various object parts. As before, we have a set of labels $L = \{\ell_1, \ell_2, \ldots \ell_k\}$, and with each node we associate a probability vector $(P_1, P_2, \ldots, P_k)$, where P_i represents the probability that label ℓ_i is the correct label for that node. In addition, we can associate with each label ℓ_i a measurement function

$$f_i : \text{image curve} \rightarrow R^{r_i}$$

which, given a curve as argument, produces an r_i component measurement vector. Then, at each node, we store not only the probability vector, but the n measurement vectors derived by applying each of the measurement functions $f_1, \ldots, f_n$ to the curve segment associated with that node. For each sequence of labels, $\ell_i \ell_j \ell_k$, we define a compatibility function

$$\alpha_{ijk} : R^{r_i} \times R^{r_j} \times R^{r_k} \rightarrow R$$

whose argument is a triple of measurement vectors. (if ijk is not a legal sequence of labels, the function is identically 0.) If we let M_n^i be the measurement vector obtained by applying measurement function f_i to the curve associated with node n, we can rewrite eq. 1 as:

$$\max_m(n,\ell_j) = \max_{\substack{a \in A \\ b \in B \\ 1 \le i,k \le |L|}} P_m(a,\ell_i) P_m(n,\ell_j) P_m(b,\ell_k) \alpha_{ijk}(M_a^i, M_n^j, M_b^k)$$

For example, suppose ℓ_i = "left wing"

ℓ_j = "nose"

ℓ_k = "right wing"

Then the measurement vector M_a^i might contain measures of the size, angle of sweep, etc. of the possible left wing defined by the curve segment associated with node a. Similarly, M^{ℓ} could contain measurements of the size, location, orientation, etc. of the possible nose at node n. The compatibility function α_{ijk} could then take into account such factors as the relative sizes of the wings, the similarity between the angle each makes with the nose, etc. Note that whereas the value of the previously define $\alpha(i,j,k)$ depended only on the label sequence, the value of the α_{ijk} defined above also depends on the properties of the curve associated with the nodes in the graph where it is applied. Because the value of α depends on its location in the graph, it is said to be a "space variant" compatibility. We might refer to α as statically space variant, in that although its value depends on its location in the graph, the value of α, given a fixed triple of nodes, remains constant throughout the relaxation iterations.

The program was modified to incorporate the space variant compatibility function defined above. The measurement functions and compatibility functions were particularly simple, yet yielded a dramatic improvement in performance. Specifically, the same measurement function was used for all four labels.

Given a curve segment, recall that we formed a two piece polygonal approximation in the process of classification. This polygon was reduced to a vector as shown in Figure 8. The 4-tuple representing this vector is the measurement vector given by applying the measurement function. Although this measurement represents very little data, it is sufficient to convey the general location, size, and orientation of the original curve segment. We then defined four compatibility functions for the four legal sequences of labels. (For an illegal sequence, $\alpha \equiv 0$). This function also was very simple. In general, we would make the range of α be $[0,1]$. However, in this case, we let the range of α be just the set $\{0,1\}$. For a legal label sequence $\ell_i \ell_j \ell_k$, α_{ijk} would be 1 if a set of criteria comparing the relative sizes and orientations of the measurement vectors were satisfied, and 0 otherwise. The result of applying the modified method to the same input image as before is shown in Figure 6. The initial graph structure consisted of the same 53 nodes as before. Now, however, the process stabilized after two iterations instead of 14, and arrived at a superior result. The final graph is displayed in Figure 7. Plane 1 no longer has an ambiguous tail. Furthermore, the noise object has been eliminated.

References

1. Wallace S. Rutkowski, Shumel Peleg, and Azriel Rosenfeld, Shape segmentation using relaxation, TR-762, Computer Science Technical Report, University of Maryland, January 1978.

2. Wallace S. Rutkowski and Azriel Rosenfeld, A comparison of corner detection techniques for chain-coded curves, TR-623, Computer Science Technical Report, University of Maryland, January 1978.

3. Wallace S. Rutkowski, Shape completion, Computer Graphics and Image Processing 9, 1979, 89-101.

4. Les Kitchen, Discrete relaxation for matching relational structures, TR-665, Computer Science Technical Report, University of Maryland, June 1978.

5. Robert Kirby, A product rule relaxation method, TR-772, Computer Science Technical Report, University of Maryland, June 1979.

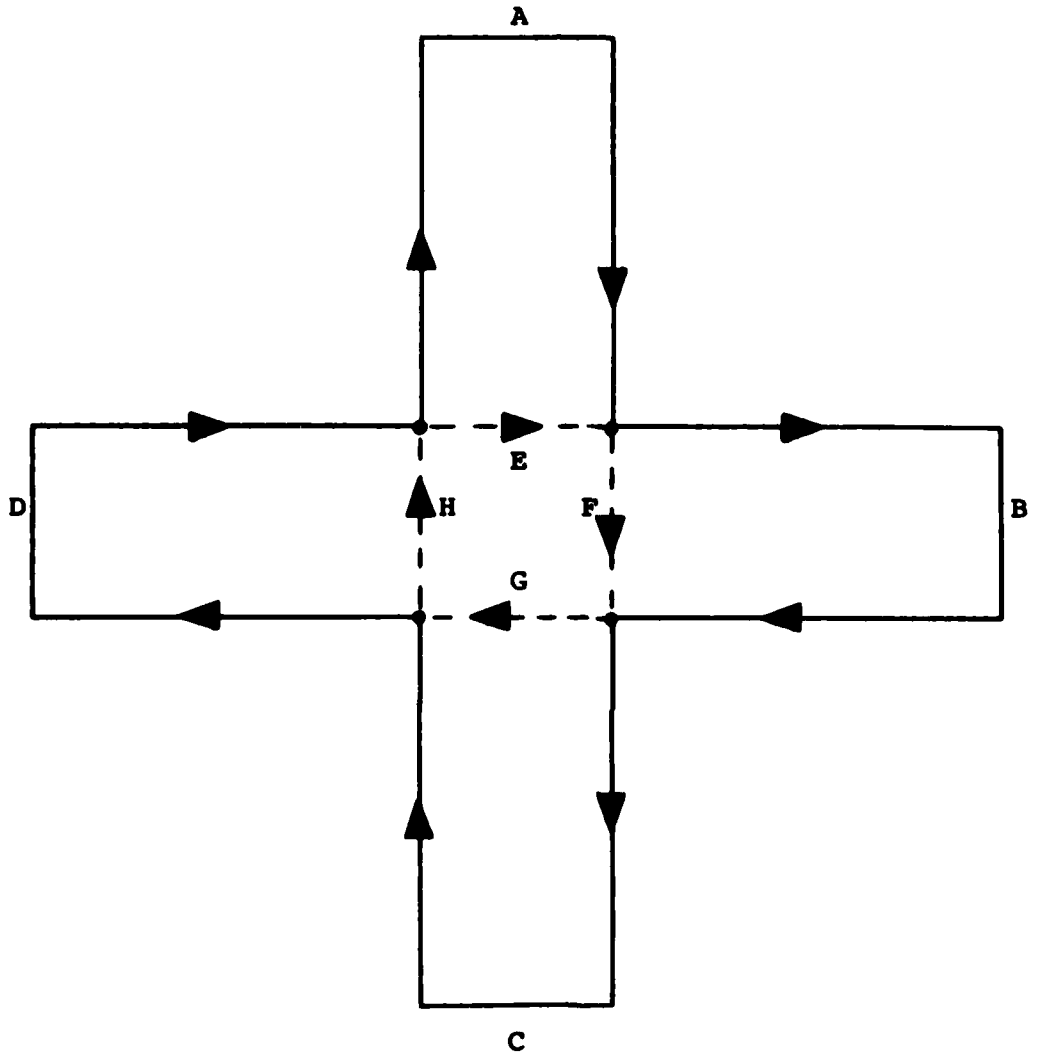


Figure 1

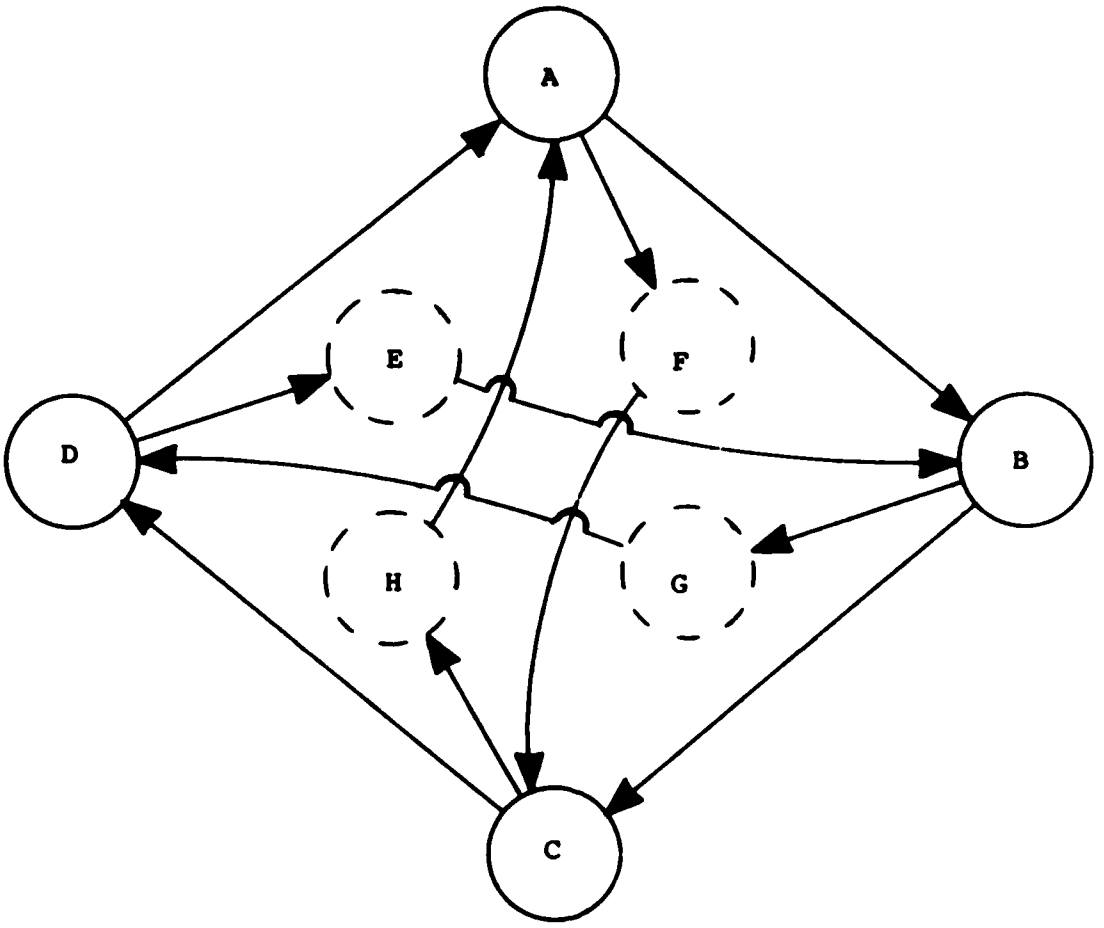
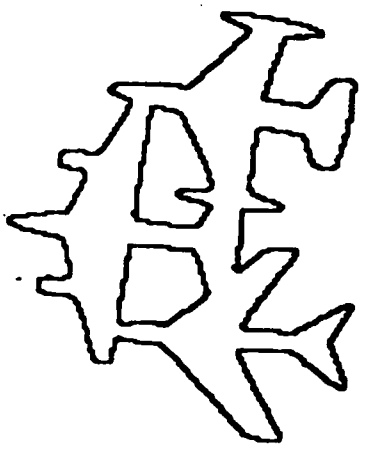
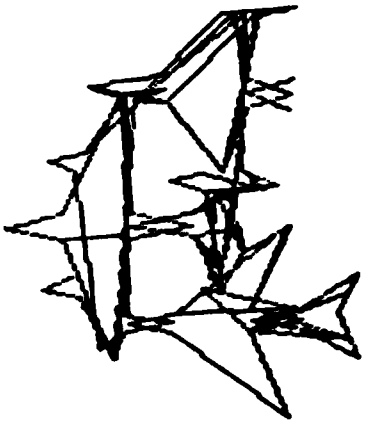


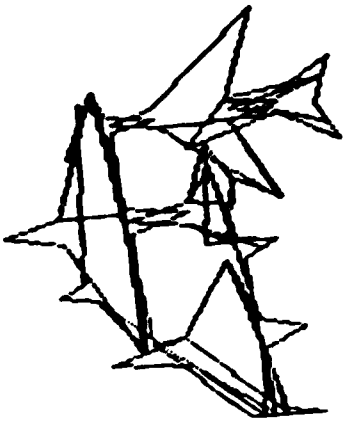
Figure 2



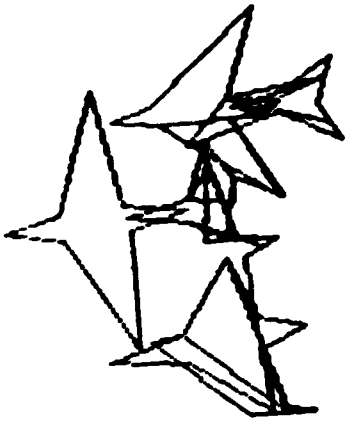
a.



b.
Iteration 0

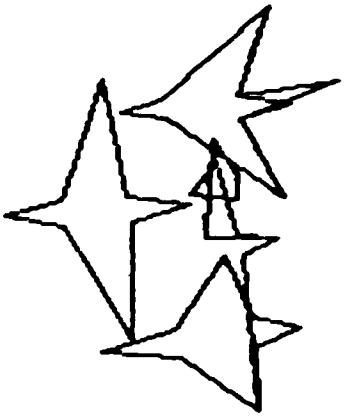


c.
Iteration 1

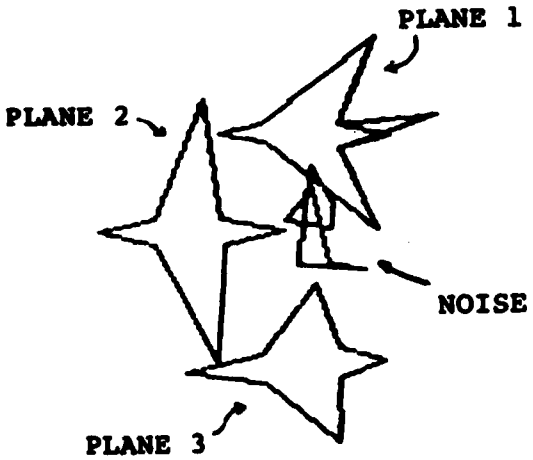


d.
Iteration 4

e.
Iteration 9



f.
Iteration 12



g.
Iteration 14

Figure 3

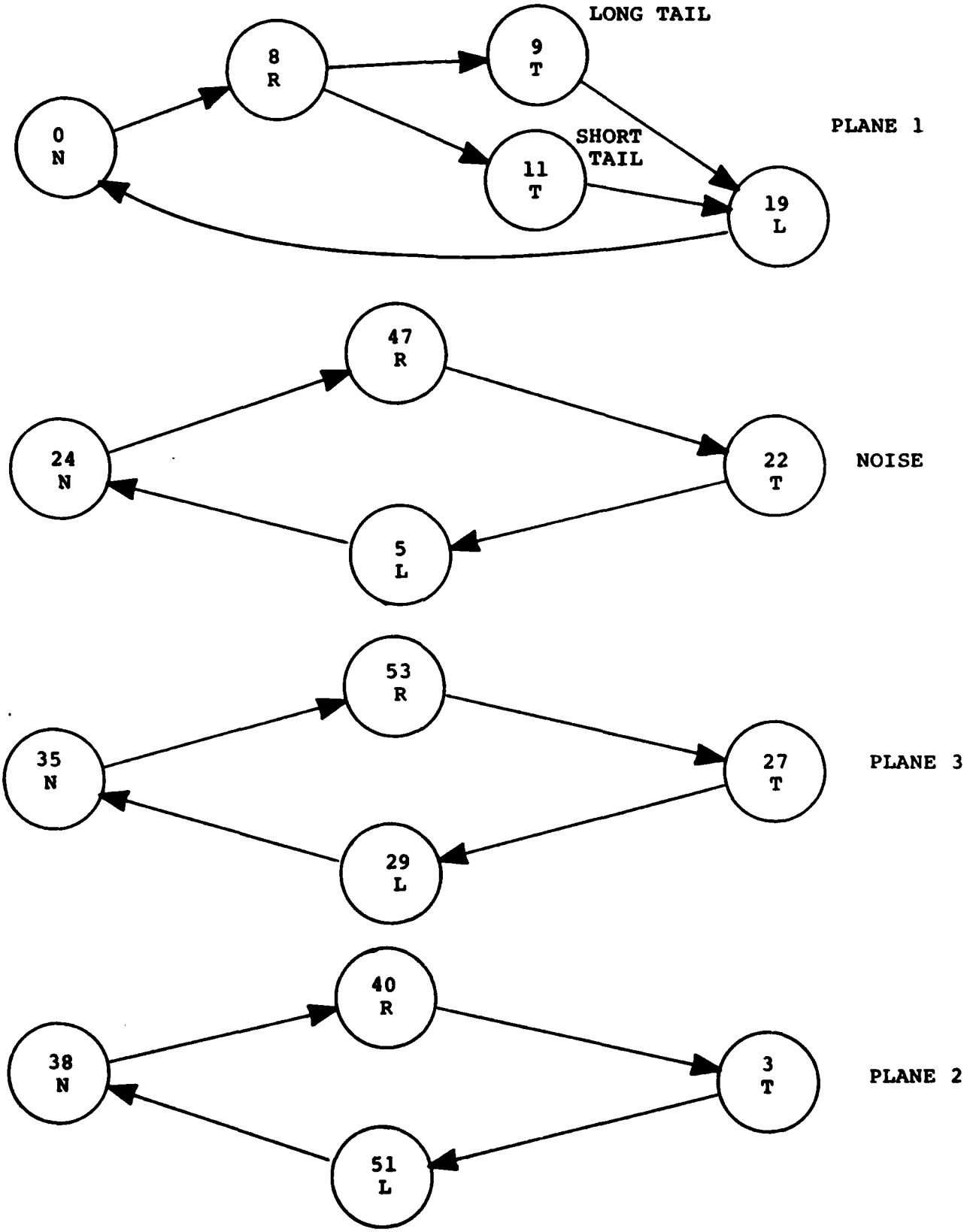
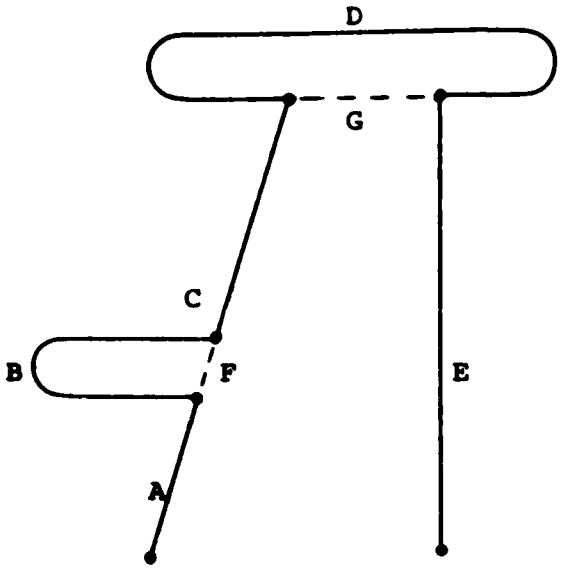
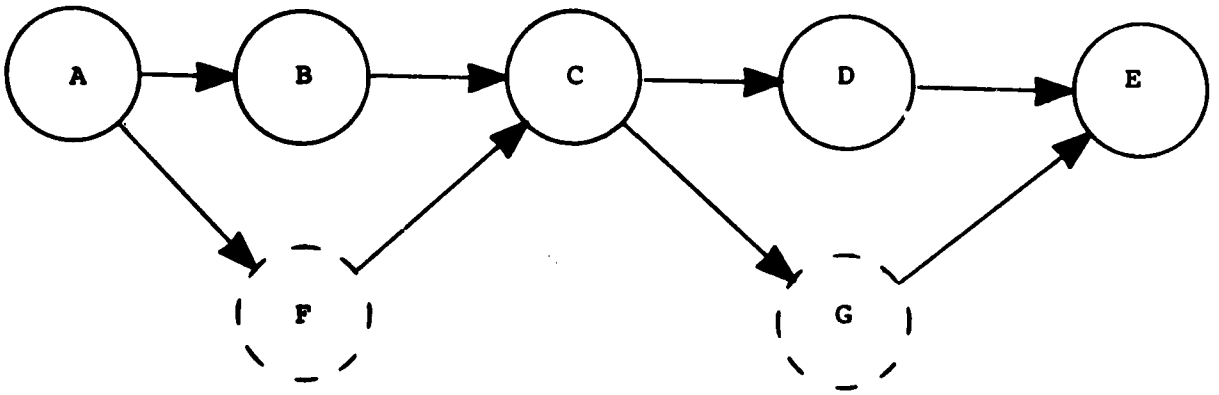

LONG TAIL
8
R
9
T
0
N
SHORT
TAIL
11
T
19
L
PLANE 1
47
R
24
N
22
T
NOISE
5
L
53
R
35
N
27
T
PLANE 3
29
L
40
R
38
N
3
T
PLANE 2
51
L

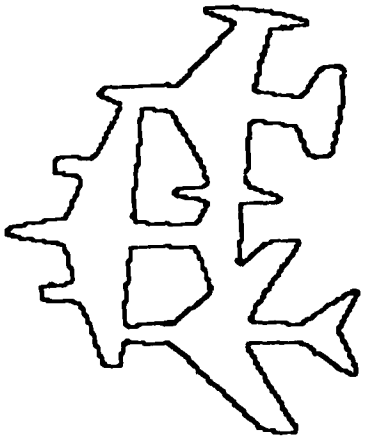
Figure 4



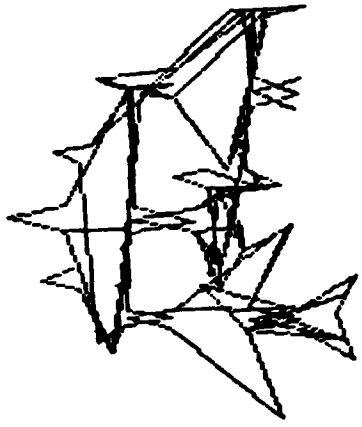


There are four paths from A to E, all of which could be classified as "right wing".

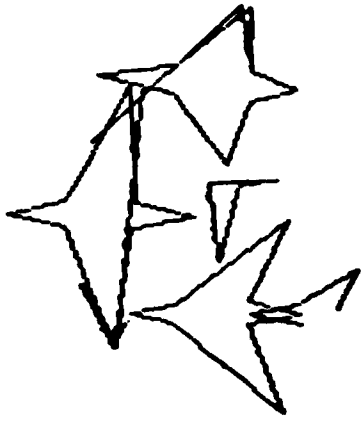
Figure 5



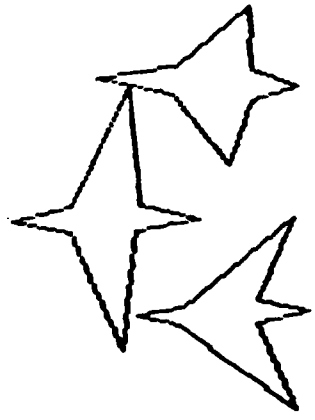
a.



b.
Iteration 0

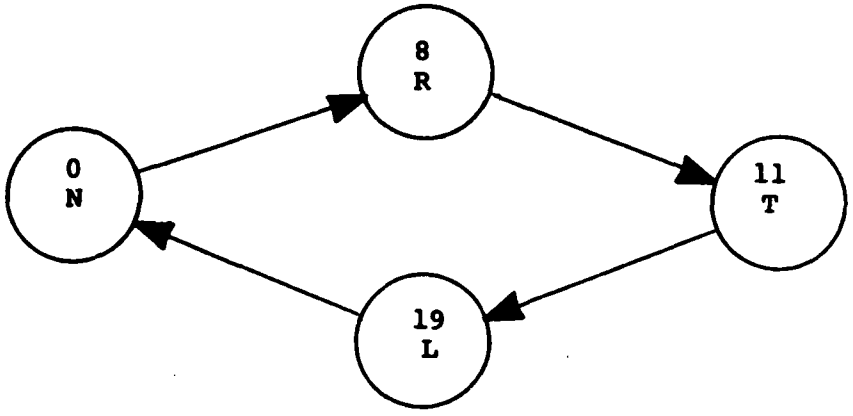


c.
Iteration 1

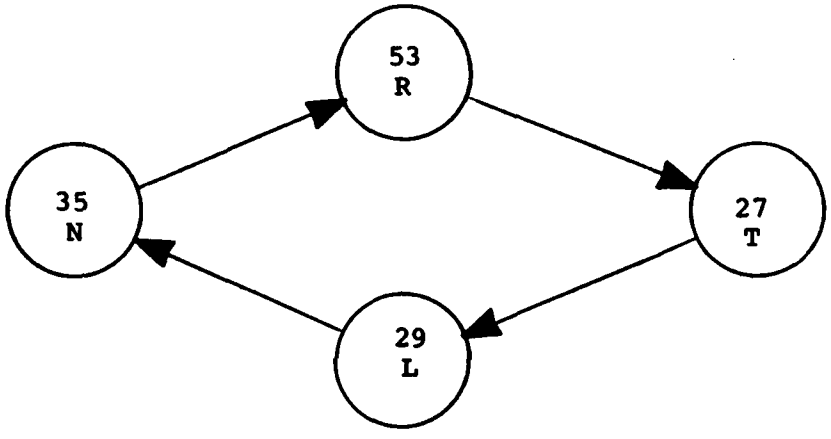


d.
Iteration 2

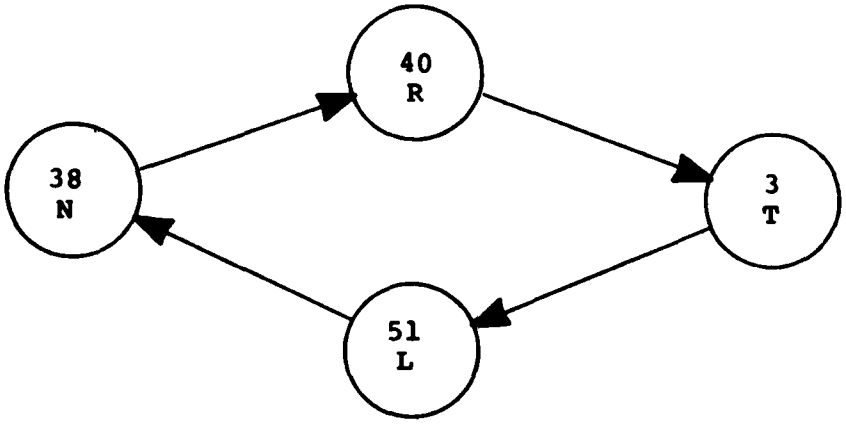
Figure 6



PLANE 1



PLANE 3



PLANE 2

Figure 7

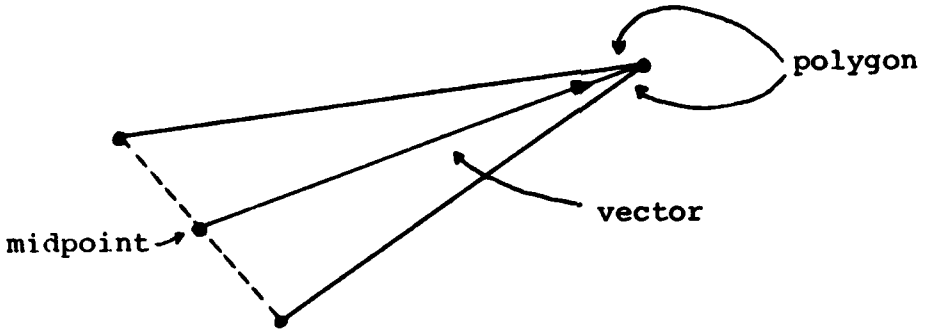


Figure 8

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. A084292	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* SHAPE SEGMENTATION USING RELAXATION, II		5. TYPE OF REPORT & PERIOD COVERED Technical
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Wallace S. Rutkowski		8. CONTRACT OR GRANT NUMBER(s) DAAG-53-76C-0138
9. PERFORMING ORGANIZATION NAME AND ADDRESS Computer Vision Laboratory, Computer Science Center, University of Maryland, College Park, MD 20742		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS U.S. Army Night Vision Laboratory Ft. Belvoir, VA 22060		12. REPORT DATE July 1979
		13. NUMBER OF PAGES 22
14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)*		15. SECURITY CLASS. *(of this report)* Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Pattern recognition
Shape recognition
Segmentation
Relaxation

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

The work begun in [1] on the application of relaxation to ambiguous shape segmentation is extended to include curve linking and gap filling. A chain coded input image is broken into segments based on a measure of local curvature. Gap completions linking pairs of segments are then proposed and represented in a graph structure. A second graph, whose nodes consist of paths in the above graph, is constructed, and the nodes of the second graph are probabilistically classified as various object parts. Relaxation is then applied

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

to increase the probability of mutually supporting classifications, and decrease the probability of unsupported decisions. A modified relaxation process using information about the size, spatial position, and orientation of the object parts yielded a high degree of disambiguation.